

Transfer Learning based Dynamic Multiobjective Optimization Algorithms

Min JIANG ¹, Zhongqiang HUANG ¹, Liming QIU ¹, Wenzhen HUANG ², Gary G. YEN ^{3,*}

1 Department of Cognitive Science and Technology, Xiamen University, China

2 Institute of Automation of the Chinese Academy of Sciences, China

3 School of Electrical and Computer Engineering, Oklahoma State University, USA

*** Corresponding authors: gyen@okstate.edu**

Abstract

One of the major distinguishing features of the dynamic multiobjective optimization problems (DMOPs) is the optimization objectives will change over time, thus tracking the varying Pareto-optimal front becomes a challenge. One of the promising solutions is reusing the “experiences” to construct a prediction model via statistical machine learning approaches. However most of the existing methods ignore the non-independent and identically distributed nature of data used to construct the prediction model. In this paper, we propose an algorithmic framework, called Tr-DMOEA, which integrates transfer learning and population-based evolutionary algorithm for solving the DMOPs. This approach takes the transfer learning method as a tool to help reuse the past experience for speeding up the evolutionary process, and at the same time, any population based multiobjective algorithms can benefit from this integration without any extensive modifications. To verify this, we incorporate the proposed approach into the development of three well-known algorithms, nondominated sorting genetic algorithm II (NSGA-II), multiobjective particle swarm optimization (MOPSO), and the regularity model-based multiobjective estimation of distribution algorithm (RM-MEDA), and

then employ twelve benchmark functions to test these algorithms as well as compare with some chosen state-of-the-art designs. The experimental results confirm the effectiveness of the proposed method through exploiting machine learning technology.

Keywords: Dynamic multi-objective optimization, Domain adaption, Dimensionality reduction, Transfer learning, Evolutionary Algorithm.

1 Introduction

One of the essential characteristics of Dynamic Multi-objective Optimization Problems (DMOPs) [18] is the objective functions will vary over time or under different environments. This underlying problem characteristic bears a significant implication for real-world applications [12]. A good example is dynamic portfolio optimization problem, which is common in deregulated electricity markets in which the operations of different power stations are controlled and coordinated to maximize profit while minimize risk. There are various uncertainties in a deregulated electricity market, including spot market prices, load obligations, and strip/option prices [48]. The values for some of these factors change over time, and it is not unusual to optimize for the market price every hour. However, the optimization approaches including population-based metaheuristic ones often find extreme difficulty to address the challenge. How to solve the DMOPs efficiently and effectively has become an important issue in evolutionary computation community [32, 13].

In recent years, a great deal of progresses has been made and different types of algorithms have been proposed. In all of these methods, one class of approaches, the prediction based, has gained much attention. This class of approaches allows evolutionary algorithm and machine learning to be seamlessly integrated. After deriving a prediction model via machine learning techniques, the evolutionary algorithms can sustain the needed performance even the environment change over time. For example, in [40], the authors proposed a memory-based evolutionary algorithm which introduced two kinds of prediction models. The first one used the linear/nonlinear regression model to predict when the environment will change while the second model was based on Markov chains which was used to forecast the changes. In [37], the authors have suggested integrating motion information into an evolutionary algorithm, such that the algorithm can track a time-changing optimum. In [44], the authors proposed a Kalman-extended genetic algorithm , and this

algorithm was developed to determine when to re-evaluate an existing individual, when to produce a new individual, and which one to re-evaluate.

The basic idea of these methods is “keeping track of good (partial) solutions in order to reuse them under periodically changing environment” [37]. If we consider this view from a statistical point of view, it seems to imply that the solutions of a dynamic optimization problems obey an identical distribution, or in other words, the solutions which are used to construct the prediction model and the solutions forecasted by the prediction model meet the Independent Identical Distribution (IID) hypothesis to some extent. However, we have to understand there is an appreciable difference between the good but out-of-dated solutions and the proper and newly generated solutions, especially under a dynamic environment.

The findings from machine learning community [34] already showed that a prediction model built by the traditional machine learning methods leaves much rooms to be desired when the training samples and the predicted samples fail to meet the IID hypothesis. Therefore, the dynamic multi-objective optimization algorithms based on traditional machine learning methods can also have a significant performance improvement by overcoming the limitation caused by the IID. Transfer learning is one of powerful tools we need.

In this paper, we argue that integrating transfer learning approaches [24] into an evolutionary algorithm can offer significant benefits for designing better Dynamic Multi-Objective Evolutionary Algorithms (DMOEAs). We adopt a domain adaptation method¹, called transfer component analysis [33], to construct a prediction model, and this model uses the gained knowledge of finding Pareto optimal solutions to generate an initial population, and based on this initial population, the optima of the changed environment can be found much efficiently and effectively. The proposed domain adaptation learning approach can be easily incorporated into any evolutionary-based multi-objective optimization algorithms.

The contribution of this research is an integration between transfer learning and classical evolutionary multi-objective optimization algorithms. This combination provides two benefits. At first, the advantages of the evolutionary algorithms are preserved in the improved design for DMOPs. Secondly, the proposed design can save computation resources via reusing the past experience which is critical for solving the DMOPs. The experiments also validate the assumption that population plays a very important role for

¹a branch of transfer learning.

tracking dynamic optima, and [15, 14] proved it from the theoretical point of view.

The rest of this paper is organized as follows: In Section 2 we will introduce some basic concepts of dynamic optimization problems first and then discuss the existing works in this field. At the beginning of Section 3 we will present some background on transfer learning, domain adaptation learning, and then introduce the transfer component analysis method in detail. After that we will propose the Transfer learning based Dynamic Multi-Objective Evolutionary optimization Algorithm, Tr-DMOEA. In Section 4 we will present the experimental results of incorporating our approach to improve three well-known MOEAs: NSGA-II, MOPSO and RM-MEDA, specifically for DMOPs and all of the algorithms were tested on the IEEE CEC 2015 benchmark problems set. In Section 5 we will draw a summary of this paper and outline the future research directions.

2 Preliminary Studies and Related Research

2.1 Dynamic Multi-objective Optimazation

Formally, a dynamic multi-objective optimization problem is defined as:

$$\begin{aligned} \text{Minimize } F(x, t) &= \langle f_1(x, t), f_2(x, t), \dots, f_M(x, t) \rangle \\ \text{s.t. } x &\in \Omega \end{aligned}$$

where $x = \langle x_1, x_2, \dots, x_n \rangle$ is the decision vector and t is the time or environment variable. $f_i(x, t) : \Omega \rightarrow \mathbb{R}$ ($i = 1, \dots, M$). $\Omega = [L_1, U_1] \times [L_2, U_2] \times \dots \times [L_n, U_n]$. $L_i, U_i \in \mathbb{R}$ and they are the lower and upper bounds of the i -th decision variable respectively.

Definition 1. [Dynamic Decision Vector Domination] *At time t , a decision vector x_1 Pareto dominate another vector x_2 , denoted by $x_1 \succ_t x_2$, if and only if:*

$$\begin{cases} \forall i = 1, \dots, M, & f_i(x_1, t) \leq f_i(x_2, t) \\ \exists i = 1, \dots, M, & f_i(x_1, t) < f_i(x_2, t) \end{cases} \quad (1)$$

Definition 2. [Dynamic Pareto-optimal Set] *Both x and x^* are decision vectors, and if a decision vector x^* is said to be nondominated at time t if and only if there is no other decision vector x such that $x \succ_t x^*$ at time t .*

The *Dynamic Pareto-Optimal Set (DPOS)* is the set of all Pareto optimal solutions at time t , that is:

$$DPOS = \{x^* \mid \nexists x, x \succ_t x^*\}.$$

Definition 3. [Dynamic Pareto-optimal Front] At time t , the *Dynamic Pareto-Optimal Front (DPOF)* is the corresponding objective vectors of the DPOS.

$$DPOF = \{F(x^*, t) \mid x^* \in DPOS\}.$$

For an ideal dynamic multiobjective algorithm, it must be able to find a set of solutions as close as possible to the changing Pareto-optimal Front, at the same time, the set of solutions should be as diverse as possible.

2.2 Related Works

Many progresses [32, 36, 2, 19] have been made in the DMOPs field in recent years, and most of the existing algorithms can be classified into the following categories: Increasing/Maintaining Diversity methods, Memory based methods, Multi-population based methods, and Prediction based methods.

The increasing diversity methods tend to add variety to the population by using a certain type of methodologies when the environment change was detected. For example, Cobb *et al.* proposed the triggered hypermutation method [10], and the basic idea of this method was that when the change was identified, the mutation rate would be increased immediately, and this would have made the converged population divergent again. This approach calls for some improvements, and one of them is that the mutation rate is in a state of uncontrolled changing during the whole process, and this ultimately results in a reduced performance of the algorithm. Therefore, Vavak *et al.* [45] presented a mutation operator, called variable local search (VLS), to address the problem. The strategy that the VLS adopted was gradually increase the mutation rate. Yen *et al.* [47] proposed a dynamic evolutionary algorithm which relocates the individuals based on their change in function value due to the change in the environment and the average sensitivities of their decision variables to the corresponding change in the objective space. This approach can avoid the drawbacks of previous methods to a certain extent.

Most of the methods in the maintaining diversity category assume that avoiding population convergence can help the algorithm track the changing optimal as soon as possible, and maintaining diversity is one of the effective

means to that end. Grefenstette [21] proposed a Random Immigrants Genetic Algorithm (RIGA), and the method replaces some individuals in the population randomly. The idea of the RIGA is that introducing new genetic materials into the population can avoid the whole population converging toward a small area in the process of evolution. However, the drawback of the primitive immigrant method was the fitness values of the introduced individuals were usually low, so the large amount of them were eliminated during the selection stage, and as a result, it is very difficult to introduce different genes into the population. For solving this problem, purpose-driven immigrant strategies have risen to prominence in the DMOPs community. Yang [51, 29] proposed the hybrid immigrants scheme, memory-based immigrants [49] and elitism-based immigrants [49], and these methods are effective for dealing with periodical changing DMOPs. However, if the knowledge about the dynamic environment is limited, they would obtain a greatly reduced efficiency.

Dynamic NSGA-II (DNSGA-II) [16] proposed by Deb *et al.* also shares the similar idea, and this method handle the DMOPs by introducing diversity when change is detected. There are two versions of the proposed DNSGA-II and there are respectively known as DNSGA-II-A and DNSGA-II-B. In the DNSGA-II-A, the population was replaced by some individuals with new randomly created solutions, while in the DNSGA-II-B diversity was guarded by replacing a percentage of the population with mutated solutions.

Memory mechanism enables evolutionary algorithms to record past information, and when it detects changes occurred, those stored information can be reused to improve the performance of the algorithm. Existing researches showed that memory-based approaches tend to be more effective on the DMOPs with periodically changed environments.

Branke [6] proposed direct memory scheme, and best solutions in the population will be saved in an archive, and when the algorithm detects a change, those saved individuals can be fetched back and be returned to the population for replacing the same number of individuals. On this basis, the different approaches were presented. In [20], the author proposed a co-evolutionary multi-objective algorithm which hybridizes competitive and cooperative mechanisms to solve the DMOPs. In this algorithm, the out-of-date archived solutions are replaced by an external population. In [46], the authors presented an algorithm called MS-MOEA to tackle the challenges of DMOPs. In the method, adaptive genetic and differential operators were used to speed up the convergence speed and a Gaussian local search opera-

tor was employed to prevent from the premature convergence. At the same time the fast hypervolume strategy was proposed to achieve a better starting population when a change occurs frequently. The above methods met some problems when the environment changes a lot. So the authors in [1] proposed an adaptive hybrid population management strategy using memory, local search and random strategies to effectively handle environment dynamics for DMOPs. The special feature of this algorithm is that it can adjust the number of memory and random solutions to be used according to the change severity.

The Multi-population strategy is considered as one efficient solution for the DMOPs, especially for the multiple peaks and the competing peaks problems. Brank *et al.* [7] proposed the self-organizing scouts method, and this method used a major population to search for optimal solution and if the major population finds a peak, then a new population would be generated to track the change of this new peak. Li and Yang [28] employed a multi-population particle swarm optimization (PSO) algorithm to solve multiple peaks problems. In their method, a population uses evolutionary programming, which shows a better global search ability, to explore the most hopeful areas in the whole search space, and at the same time, several subpopulations use the fast PSO algorithm to find the local optima. Yang [50] used hierarchical clustering technique to divide the population into different subpopulations, and the main advantage of this method is that the initial individuals of the subpopulations can be generated automatically according to the fitness landscape.

In general, a good dynamic optimization algorithm should be able to track the changing optimal solution even under high severity and frequency of changes. It must be able to reuse as much information available from previous generations to speedup the optimization search. As a result, in recent years the prediction-based DMOPs algorithms have received much attention. This class of methods predicts the state of the changing environment typically using the information that already exists and some forms of machine learning techniques, and then make a decision such that the algorithms can accommodate the changes in advance. This is one of the reasons why the prediction-based approaches can improve performance of an algorithm handling the DMOPs, compared with other types of approaches.

Bosman [5] believed that the decision made at one point would affect the optima obtained in the future, so for the dynamic optimization problems, he proposed an algorithmical framework which integrated machine learning,

statistic learning, and evolutionary computation, and this framework can effectively predict what the state of environment is going to be. In [37], the authors suggested that the state of an optimum should contain the location and the speed information, so Kalman filter technique can be used to estimate the state of the system and the error. The authors proposed an evolutionary algorithm to measure the state of the past optimum and then use the Kalman filter to obtain an estimated value of the optimum in the next time instance.

Stroud [44] proposed the Kalman-extended Genetic Algorithm (KGA), and the basic idea of the KGA was that two types of uncertainties surrounded the estimated value of an individual in a dynamical environment. The first type of the uncertainties is produced by the dynamic of the environment while the second type was related to how to evaluate the individuals. For the different situations, the KGA has two different ways to update the covariances, and uses the Kalman filter technique to predict the two uncertainties for allowing the algorithm work well in a dynamic environment.

In [53], Zhou *et al.* presented an algorithm, called Population Prediction Strategy (PPS), to predict a whole population instead of predicting some isolated points. There are two key concepts here: center point and manifold. Whenever a change is detected, the algorithm uses a sequence of center points obtained from the search progress to predict the next center point, and at the same time, the previous manifolds are used to estimate the next manifold. The main problem of this method is that, it is difficult to obtain historical information at the beginning stage, and this may lead to a poor convergence.

However, we are not exactly sure if the data we used to construct the prediction model and the data we are going to predict by the above model obey a same distribution. Conversely, the real-world applications repeatedly reminded us that, it is not wise to assume the IID hypothesis as a prerequisite, especially for the DMOPs. Unfortunately, most of the existing methods tacitly admitted that the solutions at different times have an IID structure, and we believe that this assumption is one of main reasons for the failure of existing DMOPs algorithms. After all, a poor prediction model is very likely to bring the search process to a hopeless place, which means actual results will be worse if the prediction model turn out to be inaccurate.

We believe that historical information about the Pareto-optimal front or Pareto-optimal set is very useful. A sensible use of the information will bring great benefits to track the changing Pareto-optimal front. At the same time, we must admit the rationality and the generality of the assumption of non independently and identically distributed. From these basic points of view,

we propose a framework which integrates transfer learning and evolutionary algorithms for solving the DMOPs. Two of the major advantages of the proposed approach are as follows: at first, the proposed method does not assume IID hypothesis as a prerequisite, and it is enabled to escape serious consequences of an unsuitable model. Secondly, this approach is designed to generate a population-building prediction model, so that any population-based optimization algorithms may benefit from this integration without any extensive modification.

3 Transfer Learning based Dynamic Multi-objective Optimization Algorithm

In this section we propose a transfer learning based dynamic optimization algorithm. Our motivation is the solutions of a dynamic optimization problem under different environments obey different probability distributions, and these distributions are not identical but are correlated. If we can map these different distributions into a latent space, and in this space the distributions are as “similar” as possible, then we can use the available solutions to generate an initial population, such that the solutions under new environment can be computed with low computational cost. In a spirit, this design is a reuse process of the knowledge we already obtained.

Before giving the details of the proposed approach, we need to introduce background information of the domain adaptation leaning we will use it in our design.

3.1 Transfer Component Analysis

Briefly speaking, the Domain Adaptation Learning (DAL) [3, 35], a branch of transfer learning [34], is to reuse the knowledge acquired from a source domain to perform a task in a target domain, which is related to but distinguished from the source domain. In the context of this research, a domain includes a sample space \mathcal{X} and the corresponding margin distribution $P(X)$, where $X = \{x_1, x_2, \dots, x_n\} \subseteq \mathcal{X}$. We say that two domains are different, which means they have different sample spaces and the margin distributions are different.

The researchers [4] believe that it is a promising solution using the DAL to find a good representation to decrease the difference between the distributions

of source and target domains. Gretton *et al.* [22] noted that the distance between two different distributions can be evaluated by a particular function, and in the Reproducing Kernel Hilbert Space (RKHS), the computational cost of the evaluation can be reduced. Based on this observation, Gretton *et al.* [22, 41] proposed a nonparametric distance estimation method called Maximum Mean Discrepancy (MMD) to differentiate distributions in the RKHS [43]. The MMD measures the discrepancy between two distributions by computing the difference of the mean values for the source domain and target domain. The advantages of the MMD approach is simple with a high precision.

Definition 4. (*Maximum Mean Discrepancy* [22]) : Let p and q be two Borel probability measures defined on a domain \mathcal{X} ; and $X = \{x_1, \dots, x_m\}$ and $Y = \{y_1, \dots, y_n\}$ be two observations drawn from p and q respectively. Let \mathcal{F} be a class of functions $f : \mathcal{X} \rightarrow \mathbb{R}$, then the maximum mean discrepancy (**MMD**) can be defined as:

$$\text{MMD}(\mathcal{F}, p, q) := \sup_{f \in \mathcal{F}} \left(\frac{1}{m} \sum_{i=1}^m f(x_i) - \frac{1}{n} \sum_{i=1}^n f(y_i) \right).$$

In a Reproducing Kernel Hilbert Space, f can be written as $f(x) = \langle \phi(x), f \rangle$, where $\phi(x) : \mathcal{X} \rightarrow \mathcal{H}$. So the empirical estimate of **MMD** can be rewritten as:

$$\text{MMD}(\mathcal{F}, p, q) := \left\| \frac{1}{m} \sum_{i=1}^m \phi(x_i) - \frac{1}{n} \sum_{i=1}^n \phi(y_i) \right\|_{\mathcal{H}}^2. \quad (2)$$

By using the so-called “kernel trick,” we can rewrite Equation (2) as

$$\begin{aligned} \text{MMD}(\mathcal{F}, p, q) &:= \sum_{i=1}^m \sum_{j=1}^n \text{tr} \left[\hat{K} \left(\frac{1}{m \times m} L_{ii} - \frac{1}{m \times n} L_{ij} \right. \right. \\ &\quad \left. \left. - \frac{1}{n \times m} L_{ji} + \frac{1}{n \times n} L_{jj} \right) \right] \\ &:= \text{tr}(\hat{K}L), \end{aligned} \quad (3)$$

where $\text{tr}(A)$ means the trace of the matrix A , and the matrix

$$\hat{K} = \begin{pmatrix} \hat{K}_{X,X} & \hat{K}_{X,Y} \\ \hat{K}_{Y,X} & \hat{K}_{Y,Y} \end{pmatrix} \in \mathbb{R}^{(m+n) \times (m+n)}. \quad (4)$$

$\hat{K}_{X,Y}$ is a kernel matrix with $k_{i,j} = \kappa(x_i, y_j) = \phi(x_i)^T \phi(y_j)$, where $\kappa(\cdot, \cdot)$ is a kernel function and $\phi(\cdot)$ is a feature mapping function. This matrix reflects data similarity in the domains X and Y . $\hat{K}_{X,X}$, $\hat{K}_{Y,X}$ and $\hat{K}_{Y,Y}$ have the similar meanings. Matrix L contains the coefficients to scale the matrix according to Equation (2) and its elements are as follows.

$$L(i, j) = \begin{cases} \frac{1}{m \times m}, & x_i, x_j \in X \\ \frac{1}{n \times n}, & x_i, x_j \in Y \\ -\frac{1}{m \times n}, & otherwise \end{cases}. \quad (5)$$

On the basis of the MMD, Pan *et al.* proposed a dimension reduction-based method [33] called Maximum Mean Discrepancy Embedding (MMDE) to (1) finding a low-dimensional space to reduce the difference between source and targets distributions as well as (2) to preserve the main statistical properties, maximization of data variance in the first extracted orthogonal components of the original data X and Y . In MMDE, the kernel function κ is learned (or optimized) from the data, which makes it computationally expensive, so the authors in [33] proposed other dimension reduction-based methods called Transfer Component Analysis (TCA) and its Semi-Supervised version of TCA, SSTCA, to transform the problem of learning an entire kernel matrix to a low-rank matrix W instead.

Now let us consider how to obtain the matrix W by using the TCA method. Suppose that W is a $(m + n) \times d$ matrix. For any vector x , let $\phi(x) = W^T \kappa_x \in \mathbb{R}^d$, where $\phi(\cdot)$ is a feature mapping function. Let $\kappa_x = [\kappa(x_1, x), \dots, \kappa(x_m, x), \kappa(y_1, x), \dots, \kappa(y_n, x)]^T$, and the matrix \hat{K} in Equation (4) can be transformed as follows.

$$\begin{aligned} \hat{K} &= [\phi(x_1), \dots, \phi(x_m), \phi(y_1), \dots, \phi(y_n)]^T \times \\ &\quad [\phi(x_1), \dots, \phi(x_m), \phi(y_1), \dots, \phi(y_n)] \\ &= [W^T \kappa_{x_1}, \dots, W^T \kappa_{x_m}, W^T \kappa_{y_1}, \dots, W^T \kappa_{y_n}]^T \times \\ &\quad [W^T \kappa_{x_1}, \dots, W^T \kappa_{x_m}, W^T \kappa_{y_1}, \dots, W^T \kappa_{y_n}] \\ &= [\kappa_{x_1}, \dots, \kappa_{x_m}, \kappa_{y_1}, \dots, \kappa_{y_n}]^T W W^T \\ &\quad [\kappa_{x_1}, \dots, \kappa_{x_m}, \kappa_{y_1}, \dots, \kappa_{y_n}] \\ &= K^T W W^T K \\ &= K W W^T K. \end{aligned} \quad (6)$$

Please note that the matrix K is a symmetric matrix, so $K^T = K$, and then $\text{tr}(\hat{K}L) = \text{tr}(KWW^TKL)$. According to the property of the trace of a matrix, we can rewrite Equation (3) as follows.

$$\begin{aligned}\text{MMD}(\mathcal{F}, p, q) &= \text{tr}(\hat{K}L) \\ &= \text{tr}(KWW^TKL) \\ &= \text{tr}(W^TKLKW).\end{aligned}\tag{7}$$

Now the optimization problem for the TCA algorithm can be written as follows:

$$\begin{aligned}\arg \min_W \quad & \mu \cdot \text{tr}(W^TW) + \text{tr}(W^TKLKW) \\ \text{subject to} \quad & W^TKHKW = \mathbf{I},\end{aligned}\tag{8}$$

where $H = \mathbf{I} - \frac{1}{m+n}\mathbf{1}\mathbf{1}^T$ and \mathbf{I} is a $(m+n) \times (m+n)$ identity matrix. W^TW is a regularization term. $\mathbf{1}$ is a $(m+n) \times 1$ all-ones matrix. m and n represent the numbers of samples in the source and target domains, respectively. μ is the tradeoff parameter. This optimization problem can be transformed into a trace maximization problem. According to the method presented in [38], the trace maximization problem can be solved by the Generalized Eigenvalue Decomposition (GED), and the solution is composed of the d leading eigenvectors. The pseudo-code of TCA is given in Algorithm 1.

Algorithm 1: TCA

Input: Source domain X ; target domain Y ; a kernel function $\kappa(\cdot, \cdot)$;

Output: Matrix W

- 1 Construct the Kernel Matrix \hat{K} , the Matrix L , and the Matrix H according to (4), (5) and (8) ;
 - 2 Construct the Matrix W by using the d leading eigenvectors of $(KLLK + \mu I)^{-1}KHK$;
 - 3 **return** the matrix W ;
-

3.2 Tr-DMOEA

Dynamic multi-objective optimization problem is a computationally expensive task which implies it requires a lot of computational resources to search

for the POS at a certain time. If the knowledge about the POS can be reused to predict future POFs or POSs under different environments, this usually means performance improvement as well as less computational resource consumption. So we believe that the prediction-based dynamic multi-objective optimization algorithm presents a promising solution.

However, the existing algorithms tend to ignore the assumption of non-IID, and it is obviously that the POSs under different environments obey the different distributions. This also means that those dynamic optimization algorithms based on the traditional machine learning approach leave much room for improvement. So we put forward the use of the domain adaptation technique to develop a novel DMOEA.

The approach developed is to map different distributions that the solutions obey at different times into a new latent space via the domain adaptation method. In the latent space, the MMD value of different distributions will be as small as possible while variance of the data will be kept. In other words, we will make those distributions that the solutions under different environments obey as similar as possible in the latent space, so we can map the POS we have obtained into the space, and then use those mapped solutions to construct a population which will be used to search for the POS under a new environment.

In the following Tr-DMOEA algorithm, F_t is the current dynamic optimization function assuming its POS has already been found. F_{t+1} is the POS at the next time the optimization function wants to solve. The major part of the algorithm, Tr-IPG, utilizes the POF at the time t and the transfer learning method to generate a population which can be used to find the POS at the time $t + 1$. More specifically, we take the obtained Pareto-optimal Front (POF), the POF at the time t , as a source domain; the feasible solutions of the next time, the time $t + 1$, as the target domain, and then construct a mapping function φ by using the domain adaptation approach. This mapping function will embed the distributions that the source and target domain obey separately into a latent space, and in that space the different distributions will become as small as possible. From this, we can use the POS already found to generate an initial population which can be used to search for the POS of the next moment.

Remark 1. *The numbers of the elements of X_s and Y_t are predefined. In general, more sampling often means a better result, but it also needs to pay a higher computational cost, so the decision about how many solutions needed*

Algorithm 2: Tr-IPG: Transfer Learning based Initial Population Generator

Input: The Dynamic Optimization Function $F_{t+1}(\cdot)$; the POS of the function $F_t(\cdot)$ at the time t , $POS_t = \{p_1, \dots, p_m\}$; a kernel function $\kappa(\cdot, \cdot)$.

Output: A population **Pop-init**.

```

1 Initialization;
2 For the optimization functions  $F_t(\cdot)$  and  $F_{t+1}(\cdot)$ , randomly generate
  two sets of the solutions  $X_s$  and  $Y_t$ ; /* Remark 1 */
3 Calculate the objective values of the optimization functions  $F_t(X_s)$ 
  and  $F_{t+1}(Y_t)$ ;
4  $W \leftarrow \text{TCA}(\{F_t(X_s)\}, \{F_{t+1}(Y_t)\}, \kappa)$ ;
5  $PLS \leftarrow \emptyset$ ; /* Remark 2 */
6 for every  $p \in POS_t$  do
7    $\kappa_t \leftarrow [\kappa(F_t(p_1), F_t(p)), \dots, \kappa(F_t(p_m), F_t(p))]^T$ ;
8    $\varphi(F_t(p)) \leftarrow W^T \kappa_t$ ;
9    $PLS = PLS \cup \{\varphi(p)\}$ ;
10 end
11 for every  $l \in PLS$  do
12    $x \leftarrow \underset{x}{\operatorname{argmin}} \|\varphi(F_{t+1}(x)) - l\|$  /* Remark3 */
13   Pop-init = Pop-init  $\cup \{x\}$ ;
14 end
15 return Pop-init;

```

to be produced in this step depends on the resource available.

Remark 2. *PLS is a set of the mapped solutions in the latent space.*

Remark 3. *We want to find a decision variable x , such that in the latent space, $\varphi(F_{t+1}(x))$ is closet to $l \in PLS$ in the latent space. This also means that we need to solve a single objective optimization problem here, and any single objective optimization algorithm can be applied to solve the problem. In this research, we use the Interior Point Algorithm to solve the problem.*

What the Tr-IPG algorithm outputs is a population, so it is not difficult to find that we can combine any types of the population-based optimization algorithms with the Tr-IPG to obtain a transfer learning based dynamic multi-objective evolutionary algorithm.

Algorithm 3: Tr-DMOEA: Transfer Learning based Dynamic Multi-objective Evolutionary Algorithm

Input: The Dynamic Optimization Function $F(X)$; a population based multi-objective algorithm MOA; a kernel function $\kappa(\cdot, \cdot)$.

Output: the POSs of $F(X)$.

```

1 Initialization ;
2 Use MOA to solve  $F_0(X)$  to get a  $POS_0$  ;
3 for  $t = 1$  to  $n$  do
4   | Next-Pop = Tr-IPG( $F_t(X)$ ,  $POS_{(t-1)}$ ,  $\kappa(\cdot, \cdot)$ ) ; /* When a change
   |   occurred, we use Tr-IPG to generate an init population.
   |   */
5   |  $POS_t = \mathbf{MOA}(\text{Next-Pop})$  ;
6   | return  $POS_t$  ;
7 end
```

4 Empirical Study

Practically speaking, the proposed approach is compatible with any types of population-based optimization algorithms. As a result, in our experiments, we select three well-represented algorithms with different operating principles to verify our approach. The first one is the NSGA-II [17] and it is a multiobjective evolutionary algorithm that applies nondominated sorting

and crowding distance. The second multi-objective optimization algorithm is based on particle swarm optimization and it is simply called as MOPSO [11]. The third one is the RM-MEDA [52], which is a regularity model based multiobjective estimation of distribution algorithm.

The three corresponding algorithms with the proposed transfer learning are called Tr-NSGA-II, Tr-MOPSO, and Tr-RM-MEDA, respectively for dynamic optimization. It should be noted that the original designs, NSGA-II, MOPSO, and RM-MEDA are not appropriate for dynamic optimization. It is not difficult to find that these three algorithms belong to different categories, but all of them are well-developed and proven, so it can strengthen the persuasive power and the confidence level to incorporate the proposed technology. At the same time, we also compare the new algorithms with some other state-of-art designs.

One thing we need to emphasize is that in all our experiments, the parameters are set the same. In other words, for these 12 different test functions, three different algorithms, we have used the same parameters and did not adjust the parameters, especially the parameters for the TCA method, one by one for different configurations.

4.1 Performance Metrics, Testing Functions and Settings

In this research, we use four performance metrics, the Inverted Generational Distance (IGD) and its variants, the Reactivity measure (React) and its variants, to evaluate the quality of the solutions obtained by these competing algorithms.

1. The inverted generational distance (IGD) [39] is a metric to qualify the performance of a multitobjective optimization algorithm. Let P^* be the set of uniformly distributed Pareto optimal solutions in the POF and P represent the POF obtained by the algorithm, the definition of the IGD is

$$\text{IGD}(P^*, P, C) = \frac{\sum_{v^* \in P^*} \min_{v \in P} \|v^* - v\|}{|P^*|}. \quad (9)$$

If we want the value of IGD to be as small as possible, the P should be close enough to P^* . In other words, the IGD depicts the difference between the real POF and the POF obtained by the competing algorithm.

Please note that the definition of the IGD is slightly different from the original one, and the major difference is the parameter C in Equation (9). The parameter C is a combination of the benchmark functions parameters, we call it as configuration of the benchmark functions. The configurations we used in our experiments are described in Table 2.

2. One variant of the IGD, called MIGD, can also be used to evaluate dynamic multiobjective optimization algorithms [30], and it takes the average of the IGD values in some time steps over a run as the performance metric, given by

$$\text{MIGD}(P^*, P, C) = \frac{1}{|T|} \sum_{t \in T} \text{IGD}(P_t^*, P_t, C), \quad (10)$$

where P_t^* and P^t represent the points set of the real POF and the approximate POF obtained by the algorithm at time t . We also want to evaluate those algorithms under different environments, so a novel metric, DMIGD, is defined based on the MIGD, and the definition of the DMIGD is as follows:

$$\text{DMIGD}(P^*, P, C) = \frac{1}{|E|} \sum_{C \in E} \text{IGD}(P_t^*, P_t, C), \quad (11)$$

where $|E|$ is the number of the different environments. In our experiments, we choose eight different configurations. As a result, $|E|$ equals to eight. What we want to point out is that the DMIGD can evaluate a dynamic optimization algorithm from a high-level view and it has significant difference with the MIGD since the MIGD just considers the dynamics in one environment.

3. The reactivity measure (React) [42] is used to measure the robustness of an algorithm, and its definition is as follows:

$$\text{React}_\epsilon(t, C) = \min \left\{ t' - t \mid t < t' \in \mathbb{N}, \frac{\text{acc}(t')}{\text{acc}(t)} \geq 1 - \epsilon \right\},$$

where $\text{acc}(t) = \frac{HV(\text{POF}(t))}{\max HV(\text{POF})}$ means the accuracy rate of computing the POF at the time t , and HV means the value of Hypervolume [31]. The React describes how quickly a dynamic optimization algorithm

can recover from a change, or convergence speed after changes. The value of the React is the smaller the better. We also want to evaluate the algorithms on a macro-scale, so we give two metrics based on the React,

$$\begin{aligned}\text{MReact}_\epsilon(T, C) &= \frac{1}{|T|} \sum_{t \in T} \text{React}_\epsilon(t, C), \\ \text{DMReact}_\epsilon(T, C) &= \frac{1}{|E|} \sum_{C \in E} \text{MReact}_\epsilon(T, C).\end{aligned}\tag{12}$$

The MReact value can be considered as an average of the React values at different time points, but under the same configuration; DMReact is an average of the MReact values under different configurations considered.

In the experiments, we take the IEEE CEC 2015 Benchmark problems set as test functions and the problem set has twelve testing functions [23]. In the definitions, the decision variables are $x = (x_1, \dots, x_n)$ and $t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor$, where n_t , τ_T , and τ_t are the severity of change, maximum number of iterations, and frequency of change respectively. Table 2 describes the different combinations of n_t , τ_t , and τ_T . Please note that, for each n_t - τ_T combination, there will be $\frac{\tau_T}{\tau_t}$ environment changes. In other words, in all of our experiments, there are altogether twenty changes for the twelve dynamic problems.

The POFs of the testing functions have different shapes and each function belongs to a certain DMOPs type. Type I means POS changes, but POF does not change; Type II means that the POS and the POF change as well; Type III means that the POF changes, but the POS does not change. From Table 1, we can find that the POF of the functions could be non-convex, convex, isolated, deceptive, continuous or discontinuous. FDA4 and FDA5 are 3-objective functions and all of the remaining are 2-objective functions.

The dimensions of the decision variables are from 10 to 30-dimension. Please note that the A, B and C values for the functions FDA5_{iso}, FDA_{dec}, DMOP2_{iso} and DMOP2_{dec} are set to G(t), 0.001 and 0.05 respectively.

In all of the experiments, we set the population size to 200 and in each generation every algorithm will generate no more than 200 solutions. As mentioned above, we make each benchmark function changes 20 times, and in every change, we let the population carry out 50 iterations.

For the TCA parameter, we set the Gaussian kernel function as the default and the expected dimensionality was set to be 20. The value of μ was set to 0.5.

Table 1: The Benchmark Functions

Name	Dim.	# of Obj.	Type
FDA4	12	2	TYPE I
FDA5	12	2	TYPE II
FDA5 _{iso}	12	3	TYPE II
FDA5 _{dec}	12	3	TYPE II
DIMP2	10	3	TYPE I
dMOP2	10	3	TYPE II
dMOP2 _{iso}	10	3	TYPE II
dMOP2 _{dec}	10	3	TYPE II
dMOP3	10	3	TYPE I
HE2	30	3	TYPE III
HE7	10	3	TYPE III
HE9	10	3	TYPE III

Table 2: Configurations of the Benchmark Functions Parameters

	n_t	T_t	T_t
C1	10	5	100
C2	10	10	200
C3	10	25	500
C4	10	50	1000
C5	1	10	200
C6	1	50	1000
C7	20	10	200
C8	20	50	1000

4.2 Experimental Results

4.2.1 IGD Metric

For each benchmark function, we perform tests under eight different configurations which are listed in Table 2. Each function will change 20 times, and after each change the algorithms would returned a POS, and then we calculated the IGD, MIGD, and DMIGD values, respectively.

The results of these experiments are described in Tables 3 to 14. These tables recorded the MIGD values of the algorithms running on different testing functions under different environments.

In these tables, the “ROC” refers to the ratio of change and we used bold face to identify those experiments where performance has been improved.

We count all the experimental results and obtain the following observations.² For the NSGA-II algorithm, the total effective rate of the Tr-NSGA-II was 78% (75/96), of which 33 testing cases increased by more than 50%, 38 increased by 5% -50% and 4 cases improved by 0 - 5%; for the MOPSO algorithm, the total effective rate was 70% (67/96), including 29 testing cases improved by more than 50%, 33 performance improved by 5% - 50%, and five improved by 0 - 5%; For the RM-MEDA, the total effective rate was 73% (70/96), including 23 of the test cases increasing by more than 50%, 39

² Please note that our experimental results are obtained without tuning the parameters one by one, and if we adjust their parameters separately for different algorithms, we have reason to believe that we can get better experimental results.

lifting 5% to 50%, and 8 improved by 0 - 5%.

These experimental results prove that the transfer learning technique can improve the performance of the existing dynamic multi-objective evolutionary algorithms. On the other hand, we would like to point out that most of the testing cases of performance degradation came from two functions - FDA5_{iso} and DMOP2_{iso}. The common characteristic of these two functions is the isolated POFs, so we suspect that the reason why the performance is poor for the two benchmark functions is inappropriate parameters settings.

We also compared the DMIGD value for the algorithms, including Multidimensional Bayesian Network based Estimation Distribution Algorithm (MBN-EDA) [27], Random immigrants strategy based multiobjective Differential evolutionary algorithm with Decomposition (RND) and the Kalman Filter prediction based DMOEA (MOEA/D-KF) [30], and the results are depicted in Table 15. The experimental results show that the transfer learning based algorithms have better performance in different situations. Even compared with chosen state-of-the-art algorithms, these transfer learning based algorithms can be much more efficient.

4.2.2 React Metric

Table 16 depicts the DMReact values of the algorithms. We found that the Tr-NSGA-II and Tr-MOPSO algorithms have shown improvements, which suggest that, at least at this parameters setting, the proposed approach can improve the adaptability of the NSGA-II and the MOPSO under dynamic environments. However the robustness of the RM-MEDA seems to be reduced, and one reason we envision is that the TCA method reduces the diversity of the solutions. So how to improve the diversity and the robustness is an interesting topic for the future research. The optimal choice of the parameter setting will be also be pursued in our future research.

5 Conclusion and Future Works

In this paper, we proposed an approach of using a transfer learning technique to enhance the performance of dynamic multi-objective evolutionary algorithms. Our basic idea is that the feasible solutions of the DMOPs at different times obey the different probability distributions, though there are some relationships between these probability distributions, but they are not

Table 3: FDA4

		NSGA-II	Tt-NSGA-II	ROC (%)	MOPSO	Tt-MOPSO	ROC (%)	RM-MEDA	Tt-RE-MEDA	ROC (%)
C1	MIGD	0.2276	0.0881	61.3188	0.0764	0.0627	18.0104	0.0678	0.0533	21.4710
	VAR	1.8246E-05	9.2507E-07		7.7743E-05	1.3804E-08		5.2059E-07	3.2465E-07	
C2	MIGD	0.2244	0.0849	62.1480	0.0760	0.0635	16.4435	0.0678	0.0524	22.7386
	VAR	4.8821E-05	2.8997E-07		7.7034E-05	3.6274E-08		3.3605E-07	4.7269E-07	
C3	MIGD	0.2123	0.0861	59.4203	0.0756	0.0636	15.8521	0.0673	0.0522	22.4152
	VAR	6.7454E-05	6.7844E-07		6.4871E-05	2.1779E-09		1.7372E-07	1.7532E-07	
C4	MIGD	0.2121	0.0857	59.6021	0.0766	0.0633	17.3353	0.0684	0.0527	22.9059
	VAR	1.3541E-05	1.2315E-06		6.4677E-05	1.3470E-07		1.6080E-07	1.5401E-07	
C5	MIGD	0.3982	0.0853	78.5738	0.0624	0.0536	14.1528	0.0673	0.0500	25.7460
	VAR	4.6983E-04	1.9502E-07		4.8391E-05	7.7322E-07		4.6476E-07	5.0149E-11	
C6	MIGD	0.4151	0.0840	79.7524	0.0616	0.0535	13.0936	0.0693	0.0501	27.7051
	VAR	5.8715E-05	3.7560E-07		5.8574E-05	1.5520E-09		2.2626E-08	6.8992E-08	
C7	MIGD	0.2003	0.0878	56.1683	0.0789	0.0639	18.9788	0.0674	0.0523	22.3155
	VAR	2.9321E-10	7.0748E-06		1.3014E-04	4.5717E-10		4.1111E-07	8.7830E-08	
C8	MIGD	0.2174	0.0847	61.0192	0.0779	0.0632	18.8763	0.0687	0.0529	23.0764
	VAR	1.6242E-04	5.8681E-07		9.0152E-05	3.5372E-08		9.4215E-09	3.9562E-09	

Table 4: FDA5

		NSGA-II	Tt-NSGA-II	ROC (%)	MOPSO	Tt-MOPSO	ROC (%)	RM-MEDA	Tt-RE-MEDA	ROC (%)
C1	MIGD	0.2400	0.1413	41.1326	0.2077	0.1182	43.0924	0.1741	0.0800	54.0663
	VAR	1.5776E-03	5.1236E-06		4.5041E-03	5.1385E-09		3.1584E-04	1.3341E-06	
C2	MIGD	0.2309	0.1437	37.7968	0.2721	0.1191	56.2091	0.1973	0.0790	59.9575
	VAR	4.9722E-05	2.2216E-04		3.4133E-02	3.8575E-06		1.7169E-03	2.6424E-07	
C3	MIGD	0.2283	0.2611	-14.3646	0.1660	0.1238	25.4400	0.2091	0.0802	61.6615
	VAR	1.9969E-07	2.0216E-04		1.3965E-04	2.5556E-06		1.5582E-03	2.8186E-07	
C4	MIGD	0.2254	0.1929	14.4507	0.1545	0.1212	21.5557	0.1863	0.0793	57.4027
	VAR	3.4071E-04	1.9428E-05		9.5470E-04	2.9989E-08		9.5869E-06	7.8247E-08	
C5	MIGD	0.6507	0.4092	37.1174	0.2825	0.1130	60.0003	0.2978	0.0698	76.5526
	VAR	1.4819E-04	1.2949E-03		1.6887E-04	1.7385E-05		4.1144E-03	4.0474E-07	
C6	MIGD	0.6709	0.4090	39.0381	0.3015	0.1270	57.8789	0.2682	0.0818	69.4866
	VAR	2.0832E-03	7.0414E-04		2.8644E-03	1.0997E-03		2.0990E-03	1.6960E-04	
C7	MIGD	0.1884	0.1499	20.3940	0.1654	0.1164	29.6483	0.1726	0.0755	56.2418
	VAR	2.4646E-05	3.2471E-08		1.0135E-04	6.6345E-05		4.3048E-05	6.5608E-10	
C8	MIGD	0.2057	0.1377	33.0929	0.1547	0.1179	23.7802	0.1661	0.0752	54.7071
	VAR	1.9567E-05	5.2939E-04		1.8381E-05	2.1885E-05		2.7332E-04	4.7222E-08	

Table 5: FDA5_{iso}

		NSGA-II	Tr-NSGA-II	ROC (%)	MOPSO	Tr-MOPSO	ROC (%)	RM-MEDA	Tr-RE-MEDA	ROC (%)
C1	MIGD	0.0990	0.1474	-48.9093	0.1143	0.1270	-11.0275	0.0666	0.0668	-0.2884
	VAR	7.4915E-06	2.1071E-03		4.7170E-09	1.2904E-04		3.7265E-09	2.6494E-08	
C2	MIGD	0.1062	0.1246	-17.3705	0.1137	0.1223	-7.5808	0.0662	0.0663	-0.0852
	VAR	1.7928E-04	1.5697E-03		1.9681E-05	3.2967E-05		5.3824E-09	3.0749E-08	
C3	MIGD	0.0999	0.1318	-31.8317	0.1169	0.1194	-2.1164	0.0669	0.0669	0.0839
	VAR	5.3260E-06	4.5014E-04		2.2940E-08	4.5508E-06		1.7117E-07	2.6062E-09	
C4	MIGD	0.0959	0.1266	-31.9813	0.1156	0.1174	-1.4932	0.0659	0.0661	-0.2978
	VAR	3.9586E-06	9.6354E-05		2.3902E-05	1.6925E-04		4.5401E-09	1.5496E-08	
C5	MIGD	0.1245	0.1496	-20.1368	0.1026	0.1155	-12.5111	0.0624	0.0613	1.8057
	VAR	1.0807E-04	2.2636E-06		2.1496E-07	7.3640E-05		4.0145E-06	1.0112E-07	
C6	MIGD	0.1098	0.1413	-28.6611	0.1053	0.1107	-5.1149	0.0611	0.0612	-0.0557
	VAR	3.2348E-06	1.4977E-04		1.6652E-05	1.4460E-10		1.2519E-08	2.1923E-09	
C7	MIGD	0.0950	0.0973	-2.4419	0.1062	0.1136	-6.9763	0.0651	0.0655	-0.6285
	VAR	1.4644E-06	1.9496E-05		2.3870E-06	1.1739E-06		5.1521E-08	1.9050E-07	
C8	MIGD	0.1081	0.1148	-6.1528	0.1100	0.1188	-8.0000	0.0653	0.0654	-0.0824
	VAR	3.9038E-04	5.5916E-04		2.5876E-06	9.8240E-05		2.1896E-09	6.6672E-09	

Table 6: FDA5_{dec}

		NSGA-II	Tr-NSGA-II	ROC (%)	MOPSO	Tr-MOPSO	ROC (%)	RM-MEDA	Tr-RE-MEDA	ROC (%)
C1	MIGD	0.3701	0.2967	19.8300	0.2068	0.1706	17.5353	0.6791	0.4096	39.6884
	VAR	1.6658E-03	4.1428E-07		1.1357E-04			6.6137E-04	3.0126E-04	
C2	MIGD	0.3640	0.3018	17.0947	0.2113	0.1465	30.6647	0.6170	0.4124	33.1499
	VAR	2.5492E-03	1.1655E-03		4.5714E-03			1.6537E-04	5.1870E-06	
C3	MIGD	0.4321	0.3194	26.0648	0.2840	0.1403	50.6122	0.6352	0.3992	37.1567
	VAR	1.9916E-02	8.9818E-06		4.2669E-05			2.5114E-03	4.9007E-04	
C4	MIGD	0.4584	0.2740	40.2280	0.1882	0.1431	23.9749	0.6393	0.4102	35.8319
	VAR	1.5253E-03	1.6655E-04		4.4340E-05			5.9788E-04	4.2766E-04	
C5	MIGD	1.1684	0.9052	22.5197	0.4569	0.4083	10.6388	0.3726	0.1268	65.9669
	VAR	6.3473E-04	4.1290E-02		4.1138E-03			6.1639E-06	1.4866E-05	
C6	MIGD	1.1842	1.1479	3.0703	0.4802	0.4237	11.7771	0.3802	0.1308	65.5900
	VAR	1.1343E-02	4.5164E-02		4.8785E-03			3.4097E-03	7.1229E-04	
C7	MIGD	0.4179	0.2011	51.8789	0.1897	0.1363	28.1784	0.6260	0.3648	41.7235
	VAR	4.7462E-03	5.6786E-05		2.8357E-05			1.7293E-04	1.9517E-05	
C8	MIGD	0.3431	0.2013	41.3357	0.1794	0.1423	20.6783	0.6739	0.3662	45.6626
	VAR	1.8925E-05	2.4026E-03		9.9454E-04			3.7766E-03	1.9394E-04	

Table 7: DIMP2

		NSGA-II	Tr-NSGA-II	ROC (%)	MOPSO	Tr-MOPSO	ROC (%)	RM-MEDA	Tr-RE-MEDA	ROC (%)
C1	MIGD	3.6287	2.5147	30.6992	2.9797	0.4010	86.5408	5.2097	4.9018	5.9113
	VAR	1.8027E-01			5.3940E-02			1.7667E-02		
C2	MIGD	4.0771	2.3481	42.4074	2.4547	0.2857	88.3599	4.8282	5.2399	-8.5262
	VAR	4.7095E-02			1.8199E+00			1.1118E-03		
C3	MIGD	4.1530	2.5097	39.5693	2.1233	0.1475	93.0529	4.8860	4.9851	-2.0293
	VAR	1.1714E-01			3.1945E-02			2.2842E-01		
C4	MIGD	3.6302	2.3739	34.6066	2.7988	0.1476	94.7278	4.6559	4.8186	-3.4948
	VAR	1.6550E-01			1.1552E+00			5.6086E-05		
C5	MIGD	4.4689	2.3552	47.2980	1.8512	0.4857	73.7642	4.9163	4.9695	-1.0823
	VAR	1.2814E-01			2.7176E-01			8.7425E-02		
C6	MIGD	3.5897	2.2009	38.6900	2.2850	0.2932	87.1703	5.0086	4.9583	1.0050
	VAR	9.8967E-03			2.5271E-01			2.7665E-01		
C7	MIGD	3.7563	2.3449	37.5743	2.0250	0.3306	83.6751	4.7298	4.8399	-2.3276
	VAR	2.4725E-04			6.1787E-03			1.6563E-03		
C8	MIGD	3.8851	2.1546	44.5419	2.4292	0.2581	89.3735	4.8794	5.1023	-4.5683
	VAR	2.5013E-01			5.2389E-01			3.5292E-02		

Table 8: DMOP2

		NSGA-II	Tr-NSGA-II	ROC (%)	MOPSO	Tr-MOPSO	ROC (%)	RM-MEDA	Tr-RE-MEDA	ROC (%)
C1	MIGD	0.1367	0.0342	74.9665	0.1472	0.0105	92.8490	0.0039	0.0366	-827.7905
	VAR	8.3806E-05	2.2300E-07		1.4966E-05	6.4208E-06		8.9592E-08	2.9249E-04	
C2	MIGD	0.1097	0.0367	66.5239	0.2784	0.0090	96.7735	0.0043	0.0066	-52.7081
	VAR	3.4897E-04	5.6430E-06		1.7974E-02	9.0819E-07		1.6877E-06	2.7414E-05	
C3	MIGD	0.1392	0.0300	78.4328	0.2780	0.0097	96.5132	0.0047	0.0029	37.5324
	VAR	1.5812E-03	2.4597E-07		2.3922E-05	2.0104E-06		4.4530E-07	4.7458E-08	
C4	MIGD	0.1234	0.0407	67.0124	0.1821	0.0092	94.9274	0.0041	0.0027	33.3552
	VAR	3.4172E-04	1.5152E-04		8.6546E-03	1.1422E-07		1.1866E-07	2.5738E-08	
C5	MIGD	0.2939	2.2819	-676.2864	0.3241	1.8752	-478.5327	18.3749	18.9887	-3.3402
	VAR	1.5190E-04	8.0239E+00		1.1403E-01	6.3602E+00		9.8567E-03	1.3102E-01	
C6	MIGD	2.3484	0.2593	88.9568	0.4015	0.0874	78.2424	18.3544	18.6601	-1.6652
	VAR	7.5218E+00	1.3435E-03		1.8818E-02	8.9815E-06		5.7303E-03	3.1135E-02	
C7	MIGD	0.0967	0.0375	61.2215	0.0252	0.0186	26.1430	0.0035	0.0028	21.0670
	VAR	5.6724E-05	9.6483E-05		6.0961E-08	2.2825E-04		6.1515E-08	7.2391E-08	
C8	MIGD	0.1136	0.0311	72.6521	0.0668	0.0104	84.3618	0.0034	0.0038	-13.0372
	VAR	8.9671E-04	1.3546E-05		4.3184E-03	3.9726E-06		5.6020E-08	3.1823E-08	

Table 9: DMOP2_{iso}

		NSGA-II	Tr-NSGA-II	ROC (%)	MOPSO	Tr-MOPSO	ROC (%)	RM-MEDA	Tr-RE-MEDA	ROC (%)
C1	MIGD	0.0026	0.0027	-4.8403	0.0048	0.0049	-0.2598	0.0019	0.0019	-0.0057
	VAR	1.4710E-12	7.0791E-10		9.0873E-09	3.2098E-10		5.4961E-14	1.0336E-11	
C2	MIGD	0.0026	0.0027	-4.4605	0.0050	0.0049	2.5526	0.0019	0.0019	-0.0855
	VAR	7.0355E-10	2.1892E-10		8.0595E-09	6.0198E-11		4.1426E-12	3.8878E-13	
C3	MIGD	0.0026	0.0027	-6.8106	0.0049	0.0048	1.6411	0.0019	0.0019	-0.0549
	VAR	1.4253E-09	2.3627E-09		9.2590E-09	1.0770E-08		5.3240E-13	3.5049E-12	
C4	MIGD	0.0026	0.0027	-3.1781	0.0051	0.0050	2.8216	0.0019	0.0019	-0.0609
	VAR	4.3446E-11	5.6923E-09		8.3902E-13	3.4667E-09		2.7556E-12	1.2409E-13	
C5	MIGD	0.1212	0.1304	-7.5446	0.1127	0.1127	-0.0139	0.1104	0.1104	0.0017
	VAR	1.4153E-05	1.2108E-05		5.5578E-08	6.6161E-10		3.0130E-12	7.3002E-12	
C6	MIGD	0.1232	0.1402	-13.7891	0.1127	0.1128	-0.0400	0.1104	0.1104	-0.0004
	VAR	1.4552E-05	2.1830E-06		4.6055E-08	1.7908E-08		7.9401E-16	2.4929E-12	
C7	MIGD	0.0025	0.0026	-2.9765	0.0048	0.0049	-0.5804	0.0019	0.0019	0.0712
	VAR	5.2352E-10	3.4636E-10		1.3988E-08	2.4884E-09		4.2732E-12	1.5004E-12	
C8	MIGD	0.0026	0.0026	-2.4833	0.0050	0.0049	3.3136	0.0019	0.0019	-0.0935
	VAR	1.1559E-10	8.7427E-10		1.6665E-08	3.8434E-08		6.8648E-12	2.1063E-12	

Table 10: DMOP2_{dec}

		NSGA-II	Tr-NSGA-II	ROC (%)	MOPSO	Tr-MOPSO	ROC (%)	RM-MEDA	Tr-RE-MEDA	ROC (%)
C1	MIGD	0.3575	0.1969	44.9115	0.3054	0.0307	89.9592	0.1104	0.0511	53.7191
	VAR	5.4036E-03	8.4656E-04		5.7504E-02	4.0371E-05		4.5859E-05	4.4133E-06	
C2	MIGD	0.4193	0.2044	51.2453	0.2230	0.0134	93.9837	0.1465	0.0523	64.3164
	VAR	3.3718E-03	1.3931E-06		7.8028E-05	1.0865E-06		2.0992E-04	4.1956E-05	
C3	MIGD	0.3923	0.2182	44.3910	0.2267	0.0138	93.9166	0.0912	0.0446	51.0528
	VAR	2.2371E-03	3.4360E-06		3.7621E-03	4.9494E-08		2.5922E-05	8.8813E-07	
C4	MIGD	0.4062	0.2432	40.1262	0.4104	0.0142	96.5336	0.1104	0.0441	60.0153
	VAR	1.5897E-03	1.2597E-03		6.8751E-02	2.4467E-06		1.6155E-04	4.1179E-05	
C5	MIGD	1.2991	0.9970	23.2553	0.6897	0.0514	92.5402	0.2322	0.2290	1.3929
	VAR	5.5417E-02	7.5395E-06		1.4321E-03	1.1253E-05		1.6904E-05	1.2066E-04	
C6	MIGD	1.5098	0.8689	42.4497	0.6992	0.0553	92.0972	0.2301	0.2213	3.8511
	VAR	2.8521E-02	1.6001E-03		2.3515E-01	2.8605E-05		3.4389E-05	7.2266E-06	
C7	MIGD	0.3701	0.2011	45.6631	0.4014	0.0115	97.1391	0.1315	0.0553	57.9290
	VAR	3.8975E-03	4.6698E-03		2.3974E-05	9.5240E-08		5.0617E-05	4.0637E-05	
C8	MIGD	0.2878	0.2142	25.5755	0.3977	0.0133	96.6628	0.1066	0.0540	49.2849
	VAR	1.7600E-03	1.8550E-04		2.0118E-02	7.5405E-07		5.8950E-05	8.2834E-05	

Table 11: DMOP3

		NSGA-II	Tt-NSGA-II	ROC (%)	MOPSO	Tt-MOPSO	ROC (%)	RM-MEDA	Tr-RE-MEDA	ROC (%)
C1	MIGD	0.1084	0.0296	72.7002	0.0105	0.0416	-296.0364	0.0033	0.0025	24.6490
	VAR	7.3738E-05	7.9556E-06		1.6573E-05	2.4358E-03		4.5813E-11		
C2	MIGD	0.1359	0.0359	73.6077	0.0143	0.0065	54.2790	0.0031	0.0024	22.0849
	VAR	2.3903E-04	2.2027E-04		4.9260E-05	5.5000E-09		6.8391E-09		
C3	MIGD	0.1486	0.0245	83.5041	0.0077	0.0062	19.3390	0.0030	0.0038	-25.1478
	VAR	1.0447E-06	3.2505E-05		3.1252E-07	8.9046E-09		3.3853E-09		
C4	MIGD	0.0922	0.0390	57.6545	0.0090	0.0064	28.1404	0.0030	0.0030	0.4896
	VAR	7.5602E-05	2.6008E-08		7.6625E-09	1.0967E-08		1.0279E-08		
C5	MIGD	0.3562	2.1887	-514.5040	0.1310	1.9281	-1371.9864	18.3638	18.4456	-0.4455
	VAR	1.0547E-03	7.7773E+00		4.5042E-04	6.4553E+00		3.4346E-03		
C6	MIGD	5.9830	5.7247	4.3171	2.0842	0.1169	94.3914	18.3348	18.4792	-0.7879
	VAR	8.8697E+00	8.1335E+00		7.5217E+00	2.9454E-06		8.5653E-03		
C7	MIGD	0.1395	0.0360	74.1808	0.0155	0.0076	51.0026	0.0029	0.0026	10.7489
	VAR	3.9562E-04	5.3965E-06		1.0151E-04	2.2629E-08		4.1482E-09		
C8	MIGD	0.1174	0.0278	76.2864	0.0086	0.0063	26.7219	0.0033	0.0024	27.4789
	VAR	1.1919E-04	1.0728E-04		1.9468E-06	2.4955E-07		3.5712E-07		

Table 12: HE2

		NSGA-II	Tt-NSGA-II	ROC (%)	MOPSO	Tt-MOPSO	ROC (%)	RM-MEDA	Tr-RE-MEDA	ROC (%)
C1	MIGD	0.1942	0.1889	2.7314	0.0795	0.0683	14.0445	0.8809	0.1162	86.8097
	VAR	1.2666E-03	1.7182E-04		1.9892E-04	1.6164E-05		9.0862E-05		
C2	MIGD	0.1522	0.1788	-17.4533	0.1019	0.0672	34.0322	0.9015	0.1171	87.0121
	VAR	1.3660E-04	6.1378E-04		2.6750E-03	1.4320E-05		3.6855E-04		
C3	MIGD	0.1572	0.1546	1.6659	0.0784	0.0697	11.1071	0.9025	0.1196	86.7433
	VAR	1.6637E-06	4.0007E-05		3.2323E-04	1.1259E-05		1.0819E-04		
C4	MIGD	0.1875	0.1765	5.8589	0.0949	0.0623	34.3376	0.8906	0.1068	88.0113
	VAR	5.7780E-04	4.0523E-05		1.7837E-03	4.6907E-07		1.1358E-04		
C5	MIGD	0.2700	0.0879	67.4516	0.0967	0.0595	38.4635	0.6971	0.0697	89.9980
	VAR	9.1661E-04	1.4581E-04		2.5397E-04	1.3812E-05		1.1198E-06		
C6	MIGD	0.3545	0.0815	77.0146	0.0894	0.0572	36.0087	0.6974	0.0726	89.5874
	VAR	1.2909E-03	1.5771E-04		2.3229E-04	9.3198E-08		2.5975E-05		
C7	MIGD	0.2078	0.1495	28.0723	0.0638	0.0650	-1.9613	0.8962	0.1038	88.4222
	VAR	4.0462E-06	1.0892E-04		9.4058E-07	6.6609E-06		7.6239E-07		
C8	MIGD	0.1531	0.1830	-19.5666	0.0728	0.0626	14.0586	0.8944	0.1082	87.9062
	VAR	2.3195E-04	5.1881E-04		4.2019E-05	2.7083E-07		3.6057E-04		

Table 13: HE7

		NSGA-II	Tr-NSGA-II	ROC (%)	MOPSO	Tr-MOPSO	ROC (%)	RM-MEDA	Tr-RE-MEDA	ROC (%)
C1	MIGD	0.0971	0.0417	57.1039	0.0612	0.0594	2.8756	0.0438	0.0354	19.3090
	VAR	1.4370E-06	1.7339E-06		2.9389E-04	3.4081E-06		1.7793E-04	4.5205E-06	
C2	MIGD	0.0988	0.0409	58.5736	0.0588	0.0608	-3.3986	0.0440	0.0344	21.8204
	VAR	1.8807E-06	2.0477E-06		7.8924E-05	6.3404E-08		1.7464E-04	1.9297E-06	
C3	MIGD	0.1012	0.0410	59.4455	0.0600	0.0611	-1.7770	0.0449	0.0353	21.4040
	VAR	1.0019E-05	2.4940E-10		5.2104E-05	1.3924E-06		1.5950E-04	1.9108E-07	
C4	MIGD	0.1036	0.0398	61.5666	0.0596	0.0598	-0.4940	0.0440	0.0351	20.2879
	VAR	3.2259E-05	1.9226E-06		3.2056E-05	3.3301E-06		1.7977E-04	1.4912E-06	
C5	MIGD	0.0779	0.0362	53.4686	0.0613	0.0528	13.9572	0.0397	0.0321	19.2266
	VAR	2.4013E-05	9.3120E-08		7.4316E-05	1.4570E-05		1.1893E-04	4.0164E-06	
C6	MIGD	0.0781	0.0326	58.3400	0.0579	0.0532	8.1117	0.0391	0.0333	14.8676
	VAR	7.6260E-06	6.2165E-08		1.5637E-05	2.0849E-05		1.4266E-04	6.3904E-06	
C7	MIGD	0.0939	0.0396	57.7739	0.0523	0.0599	-14.4283	0.0433	0.0340	21.3226
	VAR	3.4045E-05	1.9975E-06		1.2234E-05	1.8850E-06		1.3745E-04	4.2448E-07	
C8	MIGD	0.1059	0.0403	61.9792	0.0547	0.0597	-9.1221	0.0436	0.0337	22.5674
	VAR	2.5446E-06	4.2571E-06		2.3103E-06	2.1898E-06		1.5031E-04	4.0268E-06	

Table 14: HE9

		NSGA-II	Tr-NSGA-II	ROC (%)	MOPSO	Tr-MOPSO	ROC (%)	RM-MEDA	Tr-RE-MEDA	ROC (%)
C1	MIGD	0.3028	0.2580	14.7877	0.2648	0.2961	-11.8097	0.2653	0.2455	7.4804
	VAR	2.9107E-05	1.5525E-06		8.0165E-06	3.5768E-05		4.0274E-08	3.9133E-07	
C2	MIGD	0.3009	0.2617	13.0234	0.2712	0.3023	-11.4461	0.2637	0.2437	7.6087
	VAR	2.0520E-06	5.9900E-06		7.1836E-07	4.6393E-05		3.1215E-07	2.4694E-07	
C3	MIGD	0.3057	0.2548	16.6490	0.2675	0.3004	-12.2867	0.2634	0.2451	6.9490
	VAR	2.2473E-05	2.2159E-05		1.6559E-06	2.1955E-05		1.4103E-06	3.2609E-07	
C4	MIGD	0.3029	0.2635	12.9993	0.2655	0.3028	-14.0407	0.2639	0.2457	6.9055
	VAR	4.6335E-05	8.2176E-05		1.0313E-06	2.9225E-05		3.7052E-06	2.2156E-05	
C5	MIGD	0.2670	0.2270	14.9754	0.1877	0.2572	-37.0047	0.2365	0.2168	8.3525
	VAR	1.1548E-06	1.5988E-06		5.1013E-06	1.0775E-04		1.3658E-05	5.3162E-10	
C6	MIGD	0.2669	0.2279	14.6107	0.1908	0.2544	-33.2995	0.2356	0.2179	7.5252
	VAR	3.1621E-06	2.7471E-05		4.2148E-08	2.3717E-05		2.7905E-07	5.3859E-06	
C7	MIGD	0.3050	0.2558	16.1353	0.2551	0.2963	-16.1533	0.2626	0.2449	6.7409
	VAR	8.0651E-05	1.5958E-05		3.2539E-07	4.7023E-05		3.7952E-06	1.0545E-05	
C8	MIGD	0.3051	0.2579	15.4894	0.2645	0.3003	-13.5733	0.2609	0.2467	5.4267
	VAR	2.4933E-05	3.7407E-06		1.2709E-07	1.0673E-05		5.5975E-07	5.4257E-09	

Table 15: DMIGD Values of Different Algorithms

DMIGD	NSGA-II	Tr-NSGA-II	MOPSO	Tr-MoPSO	RM-MEDA	Tr-RM-MEDA	MBN-EDA	RND	MOEA/D-KF
FDA4	0.2634	0.0858	0.0732	0.0609	0.0680	0.0520	0.43	0.1698	0.1913
FDA5	0.3301	0.2306	0.2131	0.1196	0.2089	0.0776	0.51	0.5323	0.4963
FDA5_iso	0.1048	0.1292	0.1106	0.1181	0.0650	0.0649	0.64	0.1433	0.1465
FDA5_dec	0.5923	0.4559	0.2746	0.2139	0.5779	0.3275	1.27	0.5403	0.5476
DIMP2	3.8986	2.3502	2.3684	0.2937	4.8892	4.9769	6.97	17.9537	22.9536
DMOP2	0.4202	0.3439	0.2129	0.2538	4.5942	4.7130	1.4	1.4329	3.0619
DMOP2_iso	0.0325	0.0358	0.0319	0.0318	0.0290	0.0290	2.56	0.0315	0.0316
DMOP2_dec	0.6303	0.3930	0.4192	0.0254	0.1449	0.0940	2.89	9.0504	9.2188
DMOP3	0.8851	1.0133	0.2851	0.2650	4.5897	4.6177	1.38	0.0697	0.0836
HE2	0.2096	0.1501	0.0847	0.0640	0.8451	0.1017	0.83	0.0744	0.0745
HE7	0.0946	0.0390	0.0582	0.0583	0.0428	0.0342	0.21	0.1787	0.2365
HE9	0.2954	0.2508	0.2459	0.2887	0.2565	0.2383	0.36	0.3432	0.4108

identical. This is a typical non - independent identically distributed (Non-IID) problem, so classical machine learning methods are difficult to solve it.

For this reason, it is no surprise to understand why the traditional dynamic optimization algorithms which based on classical machine learning are hard to achieve a satisfactory performance. To overcome these problems, we employed the techniques from the transfer learning to develop an algorithmic framework, which will create benefits to a broad class of population-based dynamic multi-objective evolutionary algorithms.

In our approach, we consider the different probability distributions that the feasible solutions obey at the different time as the source and target domains, respectively. We hope to be able to use the gained POS from the source domain to improve computational efficiency in search for POS at the next time instance. To achieve this goal, the transfer component analysis technique is applied to find a latent space where the global feature, MMD value, of the source and target domains is as small as possible. Meanwhile the major statistical characteristics of the data, i.e., the variance will try to remain unchanged. In this way, we can use the gained POS to construct an initial population which can be used by any population-based optimization algorithm to find the POS of the next time.

We realized this idea and implemented three well-known multi-objective optimization algorithm: NSGA-II, MOPSO, and RM-MEDA. The reason why we choose these three algorithms is not only they have already been

Table 16: DMReact Value of the Algorithms

DMREACT	NSGA-II	Tr-NSGA-II	MOPSO	Tr-MOPSO	RM-MEDA	Tr-RM-MEDA
FDA4	1.9803	1.7664	1.4934	1.2895	1.5033	1.7072
FDA5	1.7204	1.5625	1.4375	1.3092	2.9638	1.6086
FDA5_iso	1.7105	1.4539	1.6283	1.7039	1.0329	1.0066
FDA5_dec	1.7928	1.8224	1.5132	1.9375	2.5000	2.5263
DIMP2	2.2697	2.2763	1.4013	1.1151	1.9243	2.0197
DMOP2	2.1645	1.9671	1.5592	1.8487	1.5789	1.7467
DMOP2_iso	1.4375	1.4836	1.4704	1.3586	1.3355	1.4605
DMOP2_dec	2.2961	2.0164	1.5461	2.1809	1.9309	2.1316
DMOP3	1.7039	1.3816	1.5329	1.3026	1.0987	1.1349
HE2	2.0428	1.9474	1.0987	1.2862	2.1086	1.4572
HE7	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
HE9	1.0000	1.0000	1.0000	1.3125	1.0000	1.0000

used extensively, but they are typical of three different metaphors: genetic algorithms, particle swarm optimization algorithms, and estimation of distribution algorithms. We compared the revised algorithms with the original designs and some chosen competing algorithms on a well-accepted benchmark set which involves twelve testing functions. Almost all the experimental results showed that introducing the transfer learning technique into the dynamic optimization algorithm can greatly improve the quality of the solutions and robustness of the algorithms.

What we also want to point out is that the proposed approach shows flexibility, that means it can be applied to rebuild a wide class of optimization algorithms make them to adapt dynamic environment but without extensive modifications. This is very important to reuse the existing algorithms. This research can be considered as a starting point, but it at least teaches us two things. At first, this work implies that different kinds of machine learning methods can bring the DMOPs merits, and secondly, it inspires us to think is it possible to reuse knowledge via the operators? Can we introduce other transfer learning technique [24] and if we can apply this new kind of optimization algorithm to solve real-world problems [9, 26, 8, 25]?

Acknowledgments

This work was supported by the National Natural Science Foundation of China (No.61003014 and No.61673328) and Foundation of Xiamen University's President (No. 20720150150). The first author also wants to thank China Scholarship Council (No. 20150631505) and Oklahoma State University for providing funding and facilities to support his research as a visiting scholar. We are very grateful to Miss Arrchana Muruganantham for providing the code of [30] and Mr. Minmin Wang for experimental assistance.

References

- [1] R. Azzouz, S. Bechikh, and L. B. Said. A dynamic multi-objective evolutionary algorithm using a change severity-based adaptive population management strategy. *Soft Comput.*, aug 2015.
- [2] R. Azzouz, S. Bechikh, and L. B. Said. Dynamic multi-objective optimization using evolutionary algorithms: a survey. In *Recent Advances in Evolutionary Multi-objective Optimization*, pages 31–70. Springer Nature, aug 2016.
- [3] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.
- [4] S. Ben-David, J. Blitzer, K. Crammer, F. Pereira, et al. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 19:137, 2007.
- [5] P. A. N. Bosman. Learning and anticipation in online dynamic optimization. In *Studies in Computational Intelligence*, pages 129–152. Springer Science mathplus Business Media, 2007.
- [6] J. Branke. Memory enhanced evolutionary algorithms for changing optimization problems. In *Proceedings of the 1999 Congress on Evolutionary Computation*. Institute of Electrical and Electronics Engineers (IEEE).
- [7] J. Branke, T. Kaussler, C. Smidt, and H. Schmeck. A Multi-population approach to dynamic optimization problems. In *Evolutionary Design*

and Manufacture, pages 299–307. Springer Science mathplus Business Media, 2000.

- [8] F. Chao, F. Chen, Y. Shen, W. He, Y. Sun, Z. Wang, C. Zhou, and M. Jiang. Robotic free writing of chinese characters via human–robot interactions. *International Journal of Humanoid Robotics*, 11(01):1450007, 2014.
- [9] F. Chao, M. H. Lee, M. Jiang, and C. Zhou. An infant development-inspired approach to robot hand-eye coordination. *International Journal of Advanced Robotic Systems*, 11, 2014.
- [10] H. G. Cobb. An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments. Technical Report Technical Report AIC-90-001, 1990.
- [11] C. C. Coello and M. S. Lechuga. Mopso: A proposal for multiple objective particle swarm optimization. In *Evolutionary Computation, 2002. CEC’02. Proceedings of the 2002 Congress on*, volume 2, pages 1051–1056. IEEE, 2002.
- [12] C. Cruz, J. R. González, and D. A. Pelta. Optimization in dynamic environments: a survey on problems methods and measures. *Soft Comput*, 15(7):1427–1448, dec 2010.
- [13] M. Daneshyari and G. G. Yen. Cultural-based particle swarm for dynamic optimisation problems. *International Journal of Systems Science*, 43(7):1284–1304, 2012.
- [14] D. Dang, T. Jansen, and P. K. Lehre. Populations can be essential in tracking dynamic optima. *Algorithmica*, pages 1–21, 2016.
- [15] D.-C. Dang, T. Jansen, and P. K. Lehre. Populations can be essential in dynamic optimisation. In *Proceedings of the 2015 on Genetic and Evolutionary Computation Conference - GECCO*,. Association for Computing Machinery (ACM), 2015.
- [16] K. Deb, U. B. R. N., and S. Karthik. Dynamic multi-objective optimization and decision-Making using modified NSGA-II: a case study on

- hydro-thermal power scheduling. In *Lecture Notes in Computer Science*, pages 803–817. Springer Science mathplus Business Media.
- [17] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.
 - [18] M. Farina, K. Deb, and P. Amato. Dynamic multiobjective optimization problems: test cases approximations, and applications. *IEEE Transactions on Evolutionary Computation*, 8(5):425–442, oct 2004.
 - [19] C.-K. Goh and K. C. Tan. Dynamic evolutionary multi-objective optimization. In *Evolutionary Multi-objective Optimization in Uncertain Environments*, pages 125–152. Springer Science mathplus Business Media.
 - [20] C.-K. Goh and K. C. Tan. A competitive-cooperative coevolutionary paradigm for dynamic multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 13(1):103–127, feb 2009.
 - [21] J. Grefenstette. Genetic algorithms for changing environments. In *Parallel Problem Solving from Nature 2*, pages 137–144. Elsevier, 1992.
 - [22] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola. A kernel method for the two-sample-problem. In *Advances in neural information processing systems*, pages 513–520, 2006.
 - [23] M. Helbig and A. Engelbrecht. Benchmark functions for cec 2015 special session and competition on dynamic multi-objective optimization. Technical report, 2015.
 - [24] M. Jiang, W. Huang, Z. Huang, and G. G. Yen. Integration of global and local metrics for domain adaptation learning via dimensionality reduction. *IEEE Transactions on Cybernetics*, early access, 2016.
 - [25] M. Jiang, Y. Yu, X. Liu, F. Zhang, and Q. Hong. Fuzzy neural network based dynamic path planning. In *2012 International Conference on Machine Learning and Cybernetics*, volume 1, pages 326–330. IEEE, 2012.

- [26] M. Jiang, C. Zhou, and S. Chen. Embodied concept formation and reasoning via neural-symbolic integration. *Neurocomputing*, 74(1):113–120, 2010.
- [27] H. Karshenas, R. Santana, C. Bielza, and P. Larranaga. Multiobjective estimation of distribution algorithm based on joint modeling of objectives and variables. *IEEE Transactions on Evolutionary Computation*, 18(4):519–542, 2014.
- [28] C. Li and S. Yang. Fast Multi-Swarm Optimization for Dynamic Optimization Problems. In *2008 Fourth International Conference on Natural Computation*. Institute of Electrical and Electronics Engineers (IEEE), 2008.
- [29] M. Mavrovouniotis and S. Yang. Genetic algorithms with adaptive immigrants for dynamic environments. In *2013 IEEE Congress on Evolutionary Computation*. Institute of Electrical & Electronics Engineers (IEEE), jun 2013.
- [30] A. Muruganantham, K. C. Tan, and P. Vadakkepat. Solving the IEEE CEC 2015 dynamic benchmark problems using Kalman filter based dynamic multiobjective evolutionary algorithm. In *Proceedings in Adaptation Learning and Optimization*, pages 239–252. Springer Science math-plus Business Media, nov 2015.
- [31] A. J. Nebro, F. Luna, E. Alba, B. Dorronsoro, J. J. Durillo, and A. Beham. Abyss: Adapting scatter search to multiobjective optimization. *IEEE Transactions on Evolutionary Computation*, 12(4):439–457, 2008.
- [32] T. T. Nguyen, S. Yang, and J. Branke. Evolutionary dynamic optimization: A survey of the state of the art. *Swarm and Evolutionary Computation*, 6:1–24, oct 2012.
- [33] S. J. Pan, I. W. Tsang, J. T. Kwok, and Q. Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2011.
- [34] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on, Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

- [35] V. M. Patel, R. Gopalan, R. Li, and R. Chellappa. Visual domain adaptation: an overview of recent advances. *IEEE signal processing magazine*, 32:53–69, 2015.
- [36] C. Raquel and X. Yao. Dynamic multi-objective optimization: a survey of the state-of-the-art. In *Studies in Computational Intelligence*, pages 85–106. Springer Science mathplus Business Media, 2013.
- [37] C. Rossi, M. Abderrahim, and J. C. Díaz. Tracking moving optima using Kalman-based predictions. *Evolutionary Computation*, 16(1):1–30, mar 2008.
- [38] B. Scholkopf and K.-R. Mullert. Fisher discriminant analysis with kernels. In *Proceedings of the 1999 IEEE Signal Processing Society Workshop Neural Networks for Signal Processing IX, Madison, WI, USA*, pages 23–25, 1999.
- [39] M. R. Sierra and C. A. C. Coello. Improving pso-based multi-objective optimization using crowding, mutation and ϵ -dominance. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 505–519. Springer, 2005.
- [40] A. Simões and E. Costa. Prediction in evolutionary algorithms for dynamic environments. *Soft Comput*, 18(8):1471–1497, oct 2013.
- [41] A. Smola, A. Gretton, L. Song, and B. Schölkopf. A hilbert space embedding for distributions. In *Algorithmic Learning Theory*, pages 13–31. Springer, 2007.
- [42] M. C. Sola. *Parallel processing for dynamic multi-objective optimization*. PhD thesis, Ph. D. thesis, Universidad de Granada, Dept. of Computer Architecture and Computer Technology, Universidad de Granada, Spain, 2010.
- [43] I. Steinwart. On the influence of the kernel on the consistency of support vector machines. *The Journal of Machine Learning Research*, 2:67–93, 2002.
- [44] P. Stroud. Kalman-extended genetic algorithm for search in nonstationary environments with noisy fitness evaluations. *IEEE Transactions on Evolutionary Computation*, 5(1):66–77, 2001.

- [45] F. Vavak, T. C. Fogarty, and K. Jukes. A genetic algorithm with variable range of local search for tracking changing environments. In *Parallel Problem Solving from Nature — PPSN IV*, pages 376–385. Springer Science mathplus Business Media, 1996.
- [46] Y. Wang and B. Li. Multi-strategy ensemble evolutionary algorithm for dynamic multi-objective optimization. *Memetic Comp.*, 2(1):3–24, sep 2009.
- [47] Y. Woldesenbet and G. Yen. Dynamic evolutionary algorithm with variable relocation. *IEEE Transactions on Evolutionary Computation*, 13(3):500–513, jun 2009.
- [48] J. Xu, P. B. Luh, F. B. White, E. Ni, and K. Kasiviswanathan. Power portfolio optimization in deregulated electricity markets with risk management. *IEEE Transactions on Power Systems*, 21(4):1653–1662, 2006.
- [49] S. Yang. Genetic algorithms with memory- and elitism-based immigrants in dynamic environments. *Evolutionary Computation*, 16(3):385–416, sep 2008.
- [50] S. Yang and C. Li. A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. *IEEE Transactions on Evolutionary Computation*, 14(6):959–974, dec 2010.
- [51] S. Yang and R. Tinós. A hybrid immigrants scheme for genetic algorithms in dynamic environments. *International Journal of Automation and Computing*, 4(3):243–254, jul 2007.
- [52] Q. Zhang, A. Zhou, and Y. Jin. Rm-meda: A regularity model-based multiobjective estimation of distribution algorithm. *IEEE Transactions on Evolutionary Computation*, 12(1):41–63, 2008.
- [53] A. Zhou, Y. Jin, and Q. Zhang. A population prediction strategy for evolutionary dynamic multiobjective optimization. *IEEE Transactions on Cybernetics*, 44(1):40–53, jan 2014.